本书仅提供部分阅读，如需完整版，请联系QQ: 2404062482
提供各种IT类书籍pdf下载，如有需要，请QQ:2404062482
注：链接至淘宝，不喜者勿入！整理那么多资料也不容易，请多多见谅！非诚勿扰！
点击购买完整版

VISIT…

TURING 图灵程序设计丛书

MongoDB: The Definitive Guide

第2版

MongoDB

权威指南

[美] *Kristina Chodorow* 著
邓强 王明辉 译

O'REILLY®

人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书

MongoDB权威指南（第2版）

MongoDB: The Definitive Guide Second Edition

[美] Kristina Chodorow 著
邓强　王明辉 译

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

a, Inc.授权人民邮电出版社出版

人民邮电出版社
北　京

图书在版编目（CIP）数据

MongoDB权威指南 : 第2版 / (美) 霍多罗夫(Chodorow,K.) 著 ; 邓强, 王明辉译. -- 北京 : 人民邮电出版社, 2014.1
（图灵程序设计丛书）
书名原文: MongoDB:the definitive guide, second edition
ISBN 978-7-115-34108-2

Ⅰ. ①M… Ⅱ. ①霍… ②邓… ③王… Ⅲ. ①关系数据库系统 Ⅳ. ①TP311.138

中国版本图书馆CIP数据核字(2013)第302154号

内 容 提 要

与传统的关系型数据库不同，MongoDB 是一种面向文档的数据库。本书这一版共分为六部分，涵盖开发、管理以及部署的各个方面。第一部分展示 MongoDB 基础知识、核心概念。第二部分介绍使用 MongoDB 进行开发，包括索引的概念以及各种特殊索引和集合的用法等。第三部分讲述复制，包括副本集的相关概念、创建方法，与应用程序的交互等。第四部分讨论分片，包括分片的配置，片键的选择，集群的管理。第五部分阐述创建索引、移动和压缩数据等管理任务，以及 MongoDB 的持久数据存储。最后一部分集中说明服务器管理。

本书适合数据库开发和管理人员阅读。

◆ 著 [美] Kristina Chodorow
译 邓 强 王明辉
责任编辑 李松峰
执行编辑 李 静
责任印制 焦志炜

◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路11号
邮编 100164 电子邮件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
北京鑫正大印刷有限公司印刷

◆ 开本：800×1000 1/16
印张：25.75
字数：490千字 2014年1月第1版
印数：1－5 000册 2014年1月北京第1次印刷
著作权合同登记号 图字：01-2013-6738号

定价：79.00元

读者服务热线：(010)51095186转600 印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京崇工商广字第 0021 号

版权声明

©2013 by Kristina Chodrow.

Simplified Chinese Edition, jointly published by O'Reilly Media, Inc. and Posts & Telecom Press, 2014 Authorized translation of the English edition, 2013 O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

英文原版由 O'Reilly Media, Inc. 出版 2013。

简体中文版由人民邮电出版社出版，2014。英文原版的翻译得到 O'Reilly Media, Inc. 的授权。此简体中文版的出版和销售得到出版权和销售权的所有者 —— O'Reilly Media,Inc. 的许可。

版权所有，未得书面许可，本书的任何部分和全部不得以任何形式重制。

O'Reilly Media, Inc.介绍

O'Reilly Media 通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自 1978 年开始，O'Reilly 一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O'Reilly 的发展充满了对创新的倡导、创造和发扬光大。

O'Reilly 为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源代码峰会，以至于开源软件运动以此命名；创立了 Make 杂志，从而成为 DIY 革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O'Reilly 的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O'Reilly 现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版，在线服务或者面授课程，每一项 O'Reilly 的产品都反映了公司不可动摇的理念——信息是激发创 新的力量。

业界评论

“O'Reilly Radar 博客有口皆碑。”

——Wired

“O'Reilly 凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——Business 2.0

“O'Reilly Conference 是聚集关键思想领袖的绝对典范。”

——CRN

“一本 O'Reilly 的书就代表一个有用、有前途、需要学习的主题。”

——Irish Times

“Tim 是位特立独行的商人，他不光放眼于最长远、最广阔的视野并且切实地按照 Yogi Berra 的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去 Tim 似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——Linux Journal

序

10 年前，没人能预见互联网的发展会给关系型数据库带来如此多的挑战。在此期间，我亲身经历了在快速发展的大型互联网公司应用 MySQL 的过程。开始时只有很少的数据，一台服务器就可以了。然后就得建立备份，以便应对大量的读取和不时的宕机。用不了多长时间，就得加一个缓存层，调整所有的查询，投入更多的硬件。

最后，你会发现自己需要将数据切分到多个集群上，并重新构建大量的应用逻辑以适应这种切分。之后不久，你又会发现被自己数月前设计的数据库结构限制住了。

怎么会呢？这是因为现在集群中的数据太多，需要更改模式，会花费很长时间，也需要 DBA 投入相当多的宝贵时间。在代码中处理要简单一些，但也需要小型开发团队数月的努力。最后，你会不断地拷问自己有没有更好的方法，或者为什么没有在核心数据库服务器中内置更多此类功能。

为了应对现在 Web 应用的数据膨胀，开源社区像以往一样提供了太多的“好方法”。从内存中的键值型存储到可以使用 SQL 的 MySQL/InnoDB 变种等复杂方法，无所不有。但选择多了，做出正确的选择反而更难了。我自己就研究过其中很多种。

MongoDB 的实用性着实令人着迷。MongoDB 并不去迎合所有人的全部需求。它在功能和复杂性之间取得了很好的平衡，并且大大简化了原先十分复杂的任务。也就是说，它具备支撑今天主流 Web 应用的关键功能：索引、复制、分片、丰富的查询语法，特别灵活的数据模型。与此同时还不牺牲速度。

秉持 MongoDB 自身的风格，本书简洁明快、通俗易懂。MongoDB 新用户通过阅读第 1 章，马上就能入门，而有经验的用户则可以体验到本书的广度和权威性。对于流行的客户端 API 和高级的管理主题，如复制、备份和分片，本书都是权威参考。

根据我最近每天使用 MongoDB 的经验，我相信本书会始终不离我左右，从最初安装到进行分片或备份式集群的产品化部署，它都是我最好的助手。任何想仔细研究使用 MongoDB 的人都需要这本重要的参考书。

——Craigslist 软件工程师，Jeremy Zawodny

2010 年 8 月

前言

本书的组织结构

本书分为六个部分，涵盖了开发、管理以及部署的方方面面。

熟悉MongoDB

第 1 章将简要讲述 MongoDB 的背景：项目创立原因，希望达到的目标，选用它的理由。第 2 章接着介绍一些 MongoDB 的核心概念和术语，还有如何上手操作数据库和 shell 的相关内容。接下来两章介绍 MongoDB 开发者需要掌握的基础知识。第 3 章展示如何执行基本的写入操作，包括在不同安全和速度等级下的实现细节。第 4 章主要介绍如何查找文档和创建复杂的查询。这一章还包括如何迭代结果集和其他一些用于处理结果集的方法，比如限制结果集的数量，略过一些结果，以及对结果集排序。

使用MongoDB进行开发

第 5 章将介绍什么是索引以及如何为 MongoDB 的集合建立索引。第 6 章说明如何使用各种特殊类型的索引和集合。第 7 章展示一些利用 MongoDB 聚集数据的方法，包括计数、查找唯一值、文档分组、聚合框架和 MapReduce。这一部分的最后一章会介绍如何设计应用程序：第 8 章讲述如何更好地在应用程序中使用 MongoDB。

复制

第 9 章开始介绍复制，着重讲述如何快速在本地建立一个副本集，还会介绍一些可用选项。第 10 章涵盖了与副本集相关的一些概念。第 11 章展示了副本集与应用程序的交互。第 12 章从管理的角度介绍副本集的运行。

分片

第 13 章开始介绍分片，并通过一个例子展示如何快速地在本地进行分片。第 14 章介绍集群的组成以及设置。第 15 章介绍如何为不同的应用程序选择合适的片键。最后，第 16 章介绍分片集群的管理。

应用程序管理

接下来两章从应用程序的角度介绍 MongoDB 管理的很多方面。第 17 章讲述如何查看 MongoDB 正在进行的操作。第 18 章介绍一些管理任务，比如创建索引、移动和压缩数据。第 19 章介绍 MongoDB 的持久数据存储。

服务器管理

最后一部分集中介绍服务器管理。第 20 章将给出启动和终止 MongoDB 时的一些通用选项。第 21 章讨论在监控数据库运行时如何查看监控信息。第 22 章介绍在不同类型的部署中如何备份和恢复数据库。最后，第 23 章将介绍部署 MongoDB 时需要牢记于心的一些系统设置。

附录

附录 A 介绍了 MongoDB 的版本控制方案，以及在 Windows、OS X 和 Linux 上的安装细节。附录 B 详细说明了 MongoDB 的内部工作原理：存储引擎、数据格式和传输协议。

本书排版规范

本书使用的排版规范如下所示。

- 楷体

  用于表示新的术语。

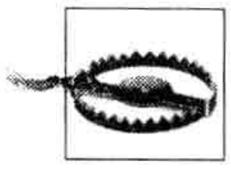
- 等宽字体

  表示程序片段，也在段落中表示程序中使用的变量、函数名、命令行实用工具、环境变量、语句和关键字等元素。

- 等宽斜体

  用户需要根据自己提供的值或由上下文确定的值进行更改的部分。

这个图标代表小窍门、建议或说明。



这个图标代表警告信息。

使用代码示例

让本书助你一臂之力。也许你要在自己的程序或文档中用到本书中的代码。除非大段大段地使用，否则不必与我们联系取得授权。例如，无需请求许可，就可以用本书中的几段代码写成一个程序。但是销售或者发布 O'Reilly 图书中代码的光盘则必须事先获得授权。引用书中的代码来回答问题也无需授权。将大段的示例代码整合到你自己的产品文档中则必须经过许可。

我们非常感谢你能标明出处，但并不强求。出处一般包括书名、作者、出版商和 ISBN，例如《MongoDB 权威指南（第 2 版）》，Kristina Chodorow 著（O'Reilly，2013）。版权所有，978-1-449-34468-9。

如果有关于使用代码的未尽事宜，可以随时与我们联系：permissions@oreilly.com。

Safari在线图书

Safari 在线图书（www.safaribooksonline.com）是应需而变的数字图书馆。它同时以图书和视频的形式出版 世界顶级技术和商务作家的专业作品。

Safari Books Online 是技术专家、软件开发人员、Web 设计师、商务人士和创意人士开展调研、解决问题、学习和认证培训的第一手资料。

对于组织团体、政府机构和个人，Safari Books Online 提供各种产品组合和灵活的定价策略。用户可通过一个功能完备的数据库检索系统访问 O'Reilly Media、Prentice Hall Professional、Addison-Wesley Professional、Microsoft Press、Sams、Que、Peachpit Press、Focal Press、Cisco Press、John Wiley & Sons、Syngress、Morgan Kaufmann、IBM Redbooks、Packt、Adobe Press、FT Press、Apress、Manning、New Riders、McGraw-Hill、Jones & Bartlett、Course Technology 以及其他几十家出版社的上千种图书、培训视频和正式出版之前的书稿。要了解 SafariBooks Online 的更多信息，我们网上见。

联系我们

请把对本书的评价和问题发给出版社。

美国：

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472

中国：

北京市西城区西直门南大街 2 号成铭大厦 C 座 807 室（100035）
奥莱利技术咨询（北京）有限公司

O'Reilly 的每一本书都有专属网页，你可以在那儿找到本书的相关信息，包括勘误表、示例代码以及其他信息。本书的网站地址是：

http://oreil.ly/mongodb_2e

对于本书的评论和技术性问题，请发送电子邮件到：

bookquestions@oreilly.com

要了解更多 O'Reilly 图书、培训课程、会议和新闻的信息，请访问以下网站：

http://www.oreilly.com

我们在 Facebook 的地址如下：

http://facebook.com/oreilly

请关注我们的 Twitter 动态：

http://twitter.com/oreillymedia

我们的 YouTube 视频地址如下：

http://www.youtube.com/oreillymedia

致谢

感谢本书的技术审稿人 Adam Comerford、Eric Milke 和 Greg Studer。他们在完善本书的过程中做出了不懈努力。感谢优秀的编辑 Ann Spencer 在本书出版的每一阶段对我的帮助。感谢我在 10gen 的各位同事，感谢他们与我分享 MongoDB 的知识和建议，同时也感谢 Eliot Horowitz 和 Dwight Merriman 启动 MongoDB 这个项目。最后要感谢 Andrew 在本书编写过程中提供的支持和建议。

目录

序 XV

前言 XVII

第一部分　MongoDB 介绍

第 1 章　MongoDB 简介 3

1.1　易于使用 3
1.2　易于扩展 4
1.3　丰富的功能 4
1.4　卓越的性能 5
1.5　小结 5

第 2 章　MongoDB 基础知识 7

2.1　文档 7
2.2　集合 8
2.2.1　动态模式 8
2.2.2　命名 9
2.3　数据库 10
2.4　启动 MongoDB 11
2.5　MongoDB shell 简介 12
2.5.1　运行 shell 12
2.5.2　MongoDB 客户端 13
2.5.3　shell 中的基本操作 14
2.6　数据类型 16

2.6.1 基本数据类型 ····· 16
2.6.2 日期 ····· 18
2.6.3 数组 ····· 18
2.6.4 内嵌文档 ····· 19
2.6.5 _id 和 ObjectId ····· 20
2.7 使用 MongoDB shell ····· 21
2.7.1 shell 小贴士 ····· 22
2.7.2 使用 shell 执行脚本 ····· 23
2.7.3 创建 .mongorc.js 文件 ····· 25
2.7.4 定制 shell 提示 ····· 26
2.7.5 编辑复合变量 ····· 26
2.7.6 集合命名注意事项 ····· 27
第 3 章 创建、更新和删除文档 ····· 29
3.1 插入并保存文档 ····· 29
3.1.1 批量插入 ····· 29
3.1.2 插入校验 ····· 30
3.2 删除文档 ····· 31
3.3 更新文档 ····· 32
3.3.1 文档替换 ····· 32
3.3.2 使用修改器 ····· 34
3.3.3 upsert ····· 45
3.3.4 更新多个文档 ····· 47
3.3.5 返回被更新的文档 ····· 48
3.4 写入安全机制 ····· 50
第 4 章 查询 ····· 53
4.1 find 简介 ····· 53
4.1.1 指定需要返回的键 ····· 54
4.1.2 限制 ····· 55
4.2 查询条件 ····· 55
4.2.1 查询条件 ····· 55
4.2.2 OR 查询 ····· 56
4.2.3 $not ····· 57
4.2.4 条件语义 ····· 57
4.3 特定类型的查询 ····· 58
4.3.1 null ····· 58
4.3.2 正则表达式 ····· 59
4.3.3 查询数组 ····· 59
4.3.4 查询内嵌文档 ····· 64

4.4 $where 查询 ····· 65
4.5 游标 ····· 67
4.5.1 limit、skip 和 sort ····· 69
4.5.2 避免使用 skip 略过大量结果 ····· 70
4.5.3 高级查询选项 ····· 72
4.5.4 获取一致结果 ····· 73
4.5.5 游标生命周期 ····· 75
4.6 数据库命令 ····· 75

第二部分 设计应用

第 5 章 索引 ····· 81
5.1 索引简介 ····· 81
5.1.1 复合索引简介 ····· 84
5.1.2 使用复合索引 ····· 90
5.1.3 $ 操作符如何使用索引 ····· 91
5.1.4 索引对象和数组 ····· 96
5.1.5 索引基数 ····· 98
5.2 使用 explain() 和 hint() ····· 98
5.3 何时不应该使用索引 ····· 103
5.4 索引类型 ····· 104
5.4.1 唯一索引 ····· 104
5.4.2 稀疏索引 ····· 106
5.5 索引管理 ····· 107
5.5.1 标识索引 ····· 108
5.5.2 修改索引 ····· 108
第 6 章 特殊的索引和集合 ····· 111
6.1 固定集合 ····· 111
6.1.1 创建固定集合 ····· 113
6.1.2 自然排序 ····· 113
6.1.3 循环游标 ····· 115
6.1.4 没有 _id 索引的集合 ····· 115
6.2 TTL 索引 ····· 116
6.3 全文本索引 ····· 116
6.3.1 搜索语法 ····· 119
6.3.2 优化全文本搜索 ····· 120
6.3.3 在其他语言中搜索 ····· 121
6.4 地理空间索引 ····· 121

6.4.1 地理空间查询的类型 ····· 122
6.4.2 复合地理空间索引 ····· 123
6.4.3 2d 索引 ····· 123
6.5 使用 GridFS 存储文件 ····· 125
6.5.1 GridFS 入门 ····· 126
6.5.2 在 MongoDB 驱动程序中使用 GridFS ····· 126
6.5.3 揭开 GridFS 的面纱 ····· 127
第 7 章 聚合 ····· 129
7.1 聚合框架 ····· 129
7.2 管道操作符 ····· 131
7.2.1 $match ····· 132
7.2.2 $project ····· 132
7.2.3 $group ····· 137
7.2.4 $unwind ····· 140
7.2.5 $sort ····· 141
7.2.6 $limit ····· 142
7.2.7 $skip ····· 142
7.2.8 使用管道 ····· 142
7.3 MapReduce ····· 143
7.3.1 示例 1：找出集合中的所有键 ····· 143
7.3.2 示例 2：网页分类 ····· 145
7.3.3 MongoDB 和 MapReduce ····· 146
7.4 聚合命令 ····· 148
7.4.1 count ····· 149
7.4.2 distinct ····· 149
7.4.3 group ····· 150
第 8 章 应用程序设计 ····· 155
8.1 范式化与反范式化 ····· 155
8.1.1 数据表示的例子 ····· 156
8.1.2 基数 ····· 159
8.1.3 好友、粉丝，以及其他的麻烦事项 ····· 160
8.2 优化数据操作 ····· 162
8.2.1 优化文档增长 ····· 162
8.2.2 删除旧数据 ····· 164
8.3 数据库和集合的设计 ····· 164
8.4 一致性管理 ····· 165
8.5 模式迁移 ····· 166
8.6 不适合使用 MongoDB 的场景 ····· 167

第三部分　复制

第 9 章　创建副本集 ····· 171

9.1　复制简介 ····· 171
9.2　建立副本集 ····· 172
9.3　配置副本集 ····· 176
9.3.1　rs 辅助函数 ····· 178
9.3.2　网络注意事项 ····· 178
9.4　修改副本集配置 ····· 178
9.5　设计副本集 ····· 180
9.6　成员配置选项 ····· 184
9.6.1　选举仲裁者 ····· 184
9.6.2　优先级 ····· 185
9.6.3　隐藏成员 ····· 186
9.6.4　延迟备份节点 ····· 187
9.6.5　创建索引 ····· 187

第 10 章　副本集的组成 ····· 189

10.1　同步 ····· 189
10.1.1　初始化同步 ····· 190
10.1.2　处理陈旧数据 ····· 193
10.2　心跳 ····· 193
10.3　选举 ····· 195
10.4　回滚 ····· 195

第 11 章　从应用程序连接副本集 ····· 201

11.1　客户端到副本集的连接 ····· 201
11.2　等待写入复制 ····· 202
11.2.1　可能导致错误的原因 ····· 203
11.2.2　"w" 的其他值 ····· 204
11.3　自定义复制保证规则 ····· 204
11.3.1　保证复制到每个数据中心的一台服务器上 ····· 204
11.3.2　保证写操作被复制到可见节点中的“大多数” ····· 206
11.3.3　创建其他规则 ····· 206
11.4　将读请求发送到备份节点 ····· 207
11.4.1　出于一致性考虑 ····· 207
11.4.2　出于负载的考虑 ····· 208
11.4.3　何时可以从备份节点读取数据 ····· 208

第 12 章　管理 ····· 211
12.1　以单机模式启动成员 ····· 211
12.2　副本集配置 ····· 212
12.2.1　创建副本集 ····· 212
12.2.2　修改副本集成员 ····· 213
12.2.3　创建比较大的副本集 ····· 213
12.2.4　强制重新配置 ····· 214
12.3　修改成员状态 ····· 215
12.3.1　把主节点变为备份节点 ····· 215
12.3.2　阻止选举 ····· 215
12.3.3　使用维护模式 ····· 215
12.4　监控复制 ····· 216
12.4.1　获取状态 ····· 216
12.4.2　复制图谱 ····· 218
12.4.3　复制循环 ····· 220
12.4.4　禁用复制链 ····· 220
12.4.5　计算延迟 ····· 221
12.4.6　调整 oplog 大小 ····· 222
12.4.7　从延迟备份节点中恢复 ····· 223
12.4.8　创建索引 ····· 224
12.4.9　在预算有限的情况下进行复制 ····· 225
12.4.10　主节点如何跟踪延迟 ····· 226
12.5　主从模式 ····· 227
12.5.1　从主从模式切换到副本集模式 ····· 228
12.5.2　让副本集模仿主从模式的行为 ····· 228

第四部分　分片

第 13 章　分片 ····· 233
13.1　分片简介 ····· 233
13.2　理解集群的组件 ····· 234
13.3　快速建立一个简单的集群 ····· 235
第 14 章　配置分片 ····· 243
14.1　何时分片 ····· 243
14.2　启动服务器 ····· 244
14.2.1　配置服务器 ····· 244
14.2.2　mongos 进程 ····· 245

14.2.3 将副本集转换为分片 ····· 245
14.2.4 增加集群容量 ····· 247
14.2.5 数据分片 ····· 247
14.3 MongoDB 如何追踪集群数据 ····· 248
14.3.1 块范围 ····· 249
14.3.2 拆分块 ····· 250
14.4 均衡器 ····· 254
第 15 章 选择片键 ····· 257
15.1 检查使用情况 ····· 257
15.2 数据分发 ····· 258
15.2.1 升序片键 ····· 258
15.2.2 随机分发的片键 ····· 261
15.2.3 基于位置的片键 ····· 262
15.3 片键策略 ····· 263
15.3.1 散列片键 ····· 264
15.3.2 GridFS 的散列片键 ····· 265
15.3.3 流水策略 ····· 266
15.3.4 多热点 ····· 267
15.4 片键规则和指导方针 ····· 270
15.4.1 片键限制 ····· 270
15.4.2 片键的势 ····· 270
15.5 控制数据分发 ····· 270
15.5.1 对多个数据库和集合使用一个集群 ····· 270
15.5.2 手动分片 ····· 272
第 16 章 分片管理 ····· 275
16.1 检查集群状态 ····· 275
16.1.1 使用 sh.status 查看集群摘要信息 ····· 275
16.1.2 检查配置信息 ····· 277
16.2 查看网络连接 ····· 282
16.2.1 查看连接统计 ····· 283
16.2.2 限制连接数量 ····· 283
16.3 服务器管理 ····· 285
16.3.1 添加服务器 ····· 285
16.3.2 修改分片的服务器 ····· 285
16.3.3 删除分片 ····· 286
16.3.4 修改配置服务器 ····· 288
16.4 数据均衡 ····· 289
16.4.1 均衡器 ····· 289

16.4.2 修改块大小……290
16.4.3 移动块……290
16.4.4 特大块……292
16.4.5 刷新配置……295

第五部分 应用管理

第 17 章 了解应用的动态……299
17.1 了解正在进行的操作……299
17.1.1 寻找有问题的操作……301
17.1.2 终止操作的执行……301
17.1.3 假象……302
17.1.4 避免幽灵操作……302
17.2 使用系统分析器……303
17.3 计算空间消耗……305
17.3.1 文档……305
17.3.2 集合……305
17.3.3 数据库……306
17.4 使用 mongotop 和 monogostat……307

第 18 章 数据管理……311
18.1 配置身份验证……311
18.1.1 身份验证基本原理……312
18.1.2 配置身份验证……313
18.1.3 身份验证的工作原理……314
18.2 建立和删除索引……315
18.2.1 在独立的服务器上建立索引……315
18.2.2 在副本集上建立索引……315
18.2.3 在分片集群上建立索引……316
18.2.4 删除索引……316
18.2.5 注意内存溢出杀手……316
18.3 预热数据……317
18.3.1 将数据库移至内存……317
18.3.2 将集合移至内存……318
18.3.3 自定义预热……318
18.4 压缩数据……320
18.5 移动集合……321
18.6 预分配数据文件……322

第 19 章　持久性……323
19.1　日记系统的用途……323
19.1.1　批量提交写入操作……324
19.1.2　设定提交时间间隔……325
19.2　关闭日记系统……325
19.2.1　替换数据文件……325
19.2.2　修复数据文件……326
19.2.3　关于 mongod.lock 文件……326
19.2.4　隐蔽的异常退出……327
19.3　MongoDB 无法保证的事项……327
19.4　检验数据损坏……327
19.5　副本集中的持久性……329

第六部分　服务器管理

第 20 章　启动和停止 MongoDB……333
20.1　从命令行启动……333
20.2　停止 MongoDB……336
20.3　安全性……337
20.3.1　数据加密……338
20.3.2　SSL 安全连接……338
20.4　日志……338
第 21 章　监控 MongoDB……341
21.1　监控内存使用状况……341
21.1.1　有关电脑内存的介绍……341
21.1.2　跟踪监测内存使用状况……342
21.1.3　跟踪监测缺页中断……343
21.1.4　减少索引树的脱靶次数……345
21.1.5　IO 延迟……345
21.1.6　跟踪监测后台刷新平均时间……346
21.2　计算工作集的大小……347
21.3　跟踪监测性能状况……349
21.4　监控副本集……352
第 22 章　备份……355
22.1　对服务器进行备份……355
22.1.1　文件系统快照……355
22.1.2　复制数据文件……356

22.1.3 使用 mongodump ····· 357
22.2 对副本集进行备份 ····· 359
22.3 对分片集群进行备份 ····· 360
22.3.1 备份和恢复整个集群 ····· 360
22.3.2 备份和恢复单独的分片 ····· 360
22.4 使用 mongooplog 进行增量备份 ····· 361
第 23 章 部署 MongoDB ····· 363
23.1 设计系统结构 ····· 363
23.1.1 选择存储介质 ····· 363
23.1.2 推荐的 RAID 配置 ····· 367
23.1.3 CPU ····· 368
23.1.4 选择操作系统 ····· 368
23.1.5 交换空间 ····· 369
23.1.6 文件系统 ····· 369
23.2 虚拟化 ····· 370
23.2.1 禁止内存过度分配 ····· 370
23.2.2 神秘的内存 ····· 370
23.2.3 处理网络磁盘的 IO 问题 ····· 371
23.2.4 使用非网络磁盘 ····· 372
23.3 系统配置 ····· 372
23.3.1 禁用 NUMA ····· 372
23.3.2 更智能地预读取数据 ····· 375
23.3.3 禁用大内存页面 ····· 376
23.3.4 选择一种磁盘调度算法 ····· 377
23.3.5 不要记录访问时间 ····· 377
23.3.6 修改限制 ····· 378
23.4 网络配置 ····· 379
23.5 系统管理 ····· 381
23.5.1 时钟同步 ····· 381
23.5.2 OOM Killer ····· 381
23.5.3 关闭定期任务 ····· 382
附录 A 安装 MongoDB ····· 383
附录 B 深入 MongoDB ····· 387

第一部分

MongoDB介绍

第1章

MongoDB简介

MongoDB是一款强大、灵活，且易于扩展的通用型数据库。它能扩展出非常多的功能，如二级索引（secondary index）、范围查询（range query）、排序、聚合（aggregation），以及地理空间索引（geospatial index）。本章涵盖了MongoDB的主要设计特点。

1.1 易于使用

MongoDB是一个面向文档（document-oriented）的数据库，而不是关系型数据库。不采用关系模型主要是为了获得更好的扩展性。当然，还有其他一些好处。

与关系型数据库相比，面向文档的数据库不再有“行”（row）的概念，取而代之的是更为灵活的“文档”（document）模型。通过在文档中嵌入文档和数组，面向文档的方法能够仅使用一条记录来表现复杂的层次关系，这与使用现代面向对象语言的开发者对数据的看法一致。

另外，不再有预定义模式（predefined schema）：文档的键（key）和值（value）不再是固定的类型和大小。由于没有固定的模式，根据需要添加或删除字段变得更容易了。通常，由于开发者能够进行快速迭代，所以开发进程得以加快。而且，实验更容易进行。开发者能尝试大量的数据模型，从中选择一个最好的。

1.2 易于扩展

应用程序数据集的大小正在以不可思议的速度增长。随着可用带宽的增长和存储器价格的下降，即使是一个小规模的应用程序，需要存储的数据量也可能大得惊人，甚至超出了很多数据库的处理能力。过去非常罕见的 T 级别数据，现在已是司空见惯了。

由于需要存储的数据量不断增长，开发者面临一个困难：应该如何扩展数据库？实质上，这是纵向扩展（scale up）和横向扩展（scale out）之间的选择。纵向扩展就是使用计算能力更强的机器，而横向扩展就是通过分区将数据分散到更多机器上。通常，纵向扩展是最省力的做法，其缺点是大型机一般都非常昂贵。而且，当数据量达到机器的物理极限时，无论花多少钱也买不到更强的机器了。另一个选择是横向扩展：要增加存储空间或提高性能，只需购买一台普通的服务器并把它添加到集群中就可以了。横向扩展既便宜又易于扩展；不过，管理 1000 台机器比管理一台机器显然要困难得多。

MongoDB 的设计采用横向扩展。面向文档的数据模型使它能很容易地在多台服务器之间进行数据分割。MongoDB 能自动处理跨集群的数据和负载，自动重新分配文档，以及将用户请求路由到正确的机器上。这样，开发者能够集中精力编写应用程序，而不需要考虑如何扩展的问题。如果一个集群需要更大的容量，只需要向集群添加新服务器，MongoDB 就会自动将现有数据向新服务器传送。

1.3 丰富的功能

MongoDB 作为一款通用型数据库，除了能够创建、读取、更新和删除数据之外，还提供一系列不断扩展的独特功能。

- *索引*（indexing）
  MongoDB 支持通用二级索引，允许多种快速查询，且提供唯一索引、复合索引、地理空间索引，以及全文索引。
- *聚合*（aggregation）
  MongoDB 支持“聚合管道”（aggregation pipeline）。用户能通过简单的片段创建复杂的聚合，并通过数据库自动优化。
- *特殊的集合类型*
  MongoDB 支持存在时间有限的集合，适用于那些将在某个时刻过期的数据，如会话（session）。类似地，MongoDB 也支持固定大小的集合，用于保存近期数据，如日志。

- 文件存储（file storage）
  MongoDB 支持一种非常易用的协议，用于存储大文件和文件元数据。

MongoDB 并不具备一些在关系型数据库中很普遍的功能，如连接（join）和复杂的多行事务（multirow transaction）。省略这些功能是出于架构上的考虑（为了得到更好的扩展性），因为在分布式系统中这两个功能难以高效地实现。

1.4 卓越的性能

MongoDB 的一个主要目标是提供卓越的性能，这很大程度上决定了 MongoDB 的设计。MongoDB 能对文档进行动态填充（dynamic padding），也能预分配数据文件以利用额外的空间来换取稳定的性能。MongoDB 把尽可能多的内存用作缓存（cache），试图为每次查询自动选择正确的索引。总之，MongoDB 在各方面的设计都旨在保持它的高性能。

虽然，MongoDB 非常强大并试图保留关系型数据库的很多特性，但它并不追求具备关系型数据库的所有功能。只要有可能，数据库服务器就会将处理和逻辑交给客户端（通过驱动程序或用户的应用程序代码来实现）。这种精简方式的设计是 MongoDB 能够实现如此高性能的原因之一。

1.5 小结

本书将详细说明 MongoDB 开发过程中的一些特定设计背后的原因和动机，借此分享 MongoDB 背后的哲学。当然，掌握 MongoDB 最好的方式是创建一个易扩展、灵活、快速的功能完备的数据存储，这也是 MongoDB 的意义所在。

第2章

MongoDB基础知识

MongoDB 非常强大但很容易上手。本章会介绍一些 MongoDB 的基本概念。

- 文档是 MongoDB 中数据的基本单元，非常类似于关系型数据库管理系统中的行，但更具表现力。
- 类似地，集合（collection）可以看作是一个拥有动态模式（dynamic schema）的表。
- MongoDB 的一个实例可以拥有多个相互独立的数据库（database），每一个数据库都拥有自己的集合。
- 每一个文档都有一个特殊的键 "_id"，这个键在文档所属的集合中是唯一的。
- MongoDB 自带了一个简单但功能强大的 JavaScript shell，可用于管理 MongoDB 的实例或数据操作。

2.1 文档

文档是 MongoDB 的核心概念。文档就是键值对的一个有序集。每种编程语言表示文档的方法不太一样，但大多数编程语言都有一些相通的数据结构，比如映射（map）、散列（hash）或字典（dictionary）。例如，在 JavaScript 里面，文档被表示为对象：

```
{"greeting" : "Hello, world!"}
```

这个文档只有一个键 "greeting"，其对应的值为 "Hello, world!"。大多数文档会比这个简单的例子复杂得多，通常会包含多个键 / 值对：

```
{"greeting" : "Hello, world!", "foo" : 3}
```

从上面的例子可以看出，文档中的值可以是多种不同的数据类型（甚至可以是一个完整的内嵌文档，详见 2.6.4 节）。在这个例子中，"greeting" 的值是一个字符串，而 "foo" 的值是一个整数。

文档的键是字符串。除了少数例外情况，键可以使用任意 UTF-8 字符。

- 键不能含有 \0（空字符）。这个字符用于表示键的结尾。
- . 和 $ 具有特殊意义，只能在特定环境下使用（后面的章节会详细说明）。通常，这两个字符是被保留的；如果使用不当的话，驱动程序会有提示。

MongoDB 不但区分类型，而且区分大小写。例如，下面的两个文档是不同的：

```
{"foo" : 3}
{"foo" : "3"}
```

下面两个文档也是不同的：

```
{"foo" : 3}
{"Foo" : 3}
```

还有一个非常重要的事项需要注意，MongoDB 的文档不能有重复的键。例如，下面的文档是非法的：

```
{"greeting" : "Hello, world!", "greeting" : "Hello, MongoDB!"}
```

文档中的键 / 值对是有序的：{"x" : 1, "y": 2} 与 {"y": 2, "x": 1} 是不同的。通常，字段顺序并不重要，无须让数据库模式依赖特定的字段顺序（MongoDB 会对字段重新排序）。在某些特殊情况下，字段顺序变得非常重要，本书将就此给出提示。

一些编程语言对文档的默认表示根本就不包含顺序问题（如：Python 中的字典、Perl 和 Ruby 1.8 中的散列）。通常，这些语言的驱动具有某些特殊的机制，可以在必要时指定文档的顺序。

2.2 集合

集合就是一组文档。如果将 MongoDB 中的一个文档比喻为关系型数据库中的一行，那么一个集合就相当于一张表。

2.2.1 动态模式

集合是动态模式的。这意味着一个集合里面的文档可以是各式各样的。例如，下面两个文档可以存储在同一个集合里面：

```
{"greeting" : "Hello, world!"}
{"foo" : 5}
```

需要注意的是，上面的文档不光值的类型不同（一个是字符串，一个是整数），它们的键也完全不同。因为集合里面可以放置任何文档，随之而来的一个问题是：还有必要使用多个集合吗？这的确值得思考：既然没有必要区分不同类型文档的模式，为什么还要使用多个集合呢？这里有几个重要的原因。

- 如果把各种各样的文档不加区分地放在同一个集合里，无论对开发者还是对管理员来说都将是噩梦。开发者要么确保每次查询只返回特定类型的文档，要么让执行查询的应用程序来处理所有不同类型的文档。如果查询博客文章时还要剔除含有作者数据的文档，这会带来很大困扰。
- 在一个集合里查询特定类型的文档在速度上也很不划算，分开查询多个集合要快得多。例如，假设集合里面一个名为 "type" 的字段用于指明文档是 skim、whole 还是 chunky monkey。那么，如果从一个集合中查询这三种类型的文档，速度会很慢。但如果将这三种不同类型的文档拆分为三个不同的集合，每次只需要查询相应的集合，速度快得多。
- 把同种类型的文档放在一个集合里，数据会更加集中。从一个只包含博客文章的集合里查询几篇文章，或者从同时包含文章数据和作者数据的集合里查出几篇文章，相比之下，前者需要的磁盘寻道操作更少。
- 创建索引时，需要使用文档的附加结构（特别是创建唯一索引时）。索引是按照集合来定义的。在一个集合中只放入一种类型的文档，可以更有效地对集合进行索引。

上面这些重要原因促使我们创建一个模式，把相关类型的文档组织在一起，尽管 MongoDB 对此并没有强制要求。

2.2.2 命名

集合使用名称进行标识。集合名可以是满足下列条件的任意 UTF-8 字符串。

- 集合名不能是空字符串（""）。
- 集合名不能包含 \0 字符（空字符），这个字符表示集合名的结束。
- 集合名不能以“system.”开头，这是为系统集合保留的前缀。例如，system.users 这个集合保存着数据库的用户信息，而 system.namespaces 集合保存着所有数据库集合的信息。
- 用户创建的集合不能在集合名中包含保留字符 '$'。因为某些系统生成的集合中包含 $，很多驱动程序确实支持在集合名里包含该字符。除非你要访问这种系统创建的集合，否则不应该在集合名中包含 $。

子集合

组织集合的一种惯例是使用“.”分隔不同命名空间的子集合。例如，一个具有博客功能的应用可能包含两个集合，分别是 blog.posts 和 blog.authors。这是为了使组织结构更清晰，这里的 blog 集合（这个集合甚至不需要存在）跟它的子集合没有任何关系。

虽然子集合没有任何特别的属性，但它们却非常有用，因而很多 MongoDB 工具都使用了子集合。

- GridFS（一种用于存储大文件的协议）使用子集合来存储文件的元数据，这样就可以与文件内容块很好地隔离开来。（第 6 章会详细介绍 GridFS。）
- 大多数驱动程序都提供了一些语法糖，用于访问指定集合的子集合。例如，在数据库 shell 中，`db.blog` 代表 blog 集合，而 `db.blog.posts` 代表 blog.posts 集合。

在 MongoDB 中，使用子集合来组织数据非常高效，值得推荐。

2.3 数据库

在 MongoDB 中，多个文档组成集合，而多个集合可以组成数据库。一个 MongoDB 实例可以承载多个数据库，每个数据库拥有 0 个或者多个集合。每个数据库都有独立的权限，即便是在磁盘上，不同的数据库也放置在不同的文件中。按照经验，我们将有关一个应用程序的所有数据都存储在同一个数据库中。要想在同一个 MongoDB 服务器上存放多个应用程序或者用户的数据，就需要使用不同的数据库。

数据库通过名称来标识，这点与集合类似。数据库名可以是满足以下条件的任意 UTF-8 字符串。

- 不能是空字符串（""）。
- 不得含有 /、\、.、"、*、<、>、:、|、?、$（一个空格）、\0（空字符）。基本上，只能使用 ASCII 中的字母和数字。
- 数据库名区分大小写，即便是在不区分大小写的文件系统中也是如此。简单起见，数据库名应全部小写。
- 数据库名最多为 64 字节。

要记住一点，数据库最终会变成文件系统里的文件，而数据库名就是相应的文件名，这是数据库名有如此多限制的原因。

另外，有一些数据库名是保留的，可以直接访问这些有特殊语义的数据库。这些数

据库如下所示。

- admin
  从身份验证的角度来讲，这是“root”数据库。如果将一个用户添加到 admin 数据库，这个用户将自动获得所有数据库的权限。再者，一些特定的服务器端命令也只能从 admin 数据库运行，如列出所有数据库或关闭服务器。

- local
  这个数据库永远都不可以复制，且一台服务器上的所有本地集合都可以存储在这个数据库中。（第 9 章会详细介绍复制及本地数据库。）

- config
  MongoDB 用于分片设置时（参见第 13 章），分片信息会存储在 config 数据库中。

把数据库名添加到集合名前，得到集合的完全限定名，即命名空间（namespace）。例如，如果要使用 cms 数据库中的 blog.posts 集合，这个集合的命名空间就是 `cms.blog.posts`。命名空间的长度不得超过 121 字节，且在实际使用中应小于 100 字节。（参考附录 B，了解 MongoDB 中集合的命名空间及内部表示的更多信息。）

2.4 启动MongoDB

通常，MongoDB 作为网络服务器来运行，客户端可连接到该服务器并执行操作。下载 MongoDB（http://www.mongodb.org/downloads）并解压，运行 `mongod` 命令，启动数据库服务器：

```
$ mongod
mongod --help for help and startup options
Thu Oct 11 12:36:48 [initandlisten] MongoDB starting : pid=2425 port=27017
    dbpath=/data/db/ 64-bit host=spock
Thu Oct 11 12:36:48 [initandlisten] db version v2.4.0, pdfile version 4.5
Thu Oct 11 12:36:48 [initandlisten] git version:
    3aaea5262d761e0bb6bfef5351cfbfca7af06ec2
Thu Oct 11 12:36:48 [initandlisten] build info: Darwin spock 11.2.0 Darwin Kernel
    Version 11.2.0: Tue Aug 9 20:54:00 PDT 2011;
    root:xnu-1699.24.8~1/RELEASE_X86_64 x86_64 BOOST_LIB_VERSION=1_48
Thu Oct 11 12:36:48 [initandlisten] options: {}
Thu Oct 11 12:36:48 [initandlisten] journal dir=/data/db/journal
Thu Oct 11 12:36:48 [initandlisten] recover : no journal files present, no
    recovery needed
Thu Oct 11 12:36:48 [websvr] admin web console waiting for connections on
    port 28017
Thu Oct 11 12:36:48 [initandlisten] waiting for connections on port 27017
```

在 Windows 系统中，执行这个命令：

```
$ mongod.exe
```

关于安装 MongoDB 的详细信息，参见附录 A。

mongod 在没有参数的情况下会使用默认数据目录 /data/db（Windows 系统中为 C:\data\db）。如果数据目录不存在或者不可写，服务器会启动失败。因此，在启动 MongoDB 前，先创建数据目录（如 mkdir -p /data/db/），以确保对该目录有写权限，这点非常重要。

启动时，服务器会打印版本和系统信息，然后等待连接。默认情况下，MongoDB 监听 27017 端口。如果端口被占用，启动将失败。通常，这是由于已经有一个 MongoDB 实例在运行了。

mongod 还会启动一个非常基本的 HTTP 服务器，监听数字比主端口号高 1000 的端口，也就是 28017 端口。这意味着，通过浏览器访问 http://localhost:28017，能获取数据库的管理信息。

中止 mongod 的运行，只须在运行着服务器的 shell 中按下 Ctrl-C。

要想了解启动和停止 MongoDB 的更多细节，参见第 20 章。

2.5 MongoDB shell简介

MongoDB 自带 JavaScript shell，可在 shell 中使用命令行与 MongoDB 实例交互。shell 非常有用，通过它可以执行管理操作，检查运行实例，亦或做其他尝试。对 MongoDB 来说，mongo shell 是至关重要的工具，其应用之广泛将体现在本书接下来的部分中。

2.5.1 运行shell

运行 mongo 启动 shell：

```
$ mongo
MongoDB shell version: 2.4.0
connecting to: test
>
```

启动时，shell 将自动连接 MongoDB 服务器，须确保 mongod 已启动。

shell 是一个功能完备的 JavaScript 解释器，可运行任意 JavaScript 程序。为说明这一点，我们运行几个简单的数学运算：

```
> x = 200
200
> x / 5;
40
```

另外，可充分利用 JavaScript 的标准库：

```
> Math.sin(Math.PI / 2);
1
> new Date("2010/1/1");
"Fri Jan 01 2010 00:00:00 GMT-0500 (EST)"
> "Hello, World!".replace("World", "MongoDB");
Hello, MongoDB!
```

再者，可定义和调用 JavaScript 函数：

```
> function factorial (n) {
... if (n <= 1) return 1;
... return n * factorial(n - 1);
... }
> factorial(5);
120
```

需要注意，可使用多行命令。shell 会检测输入的 JavaScript 语句是否完整，如没写完可在下一行接着写。在某行连续三次按下回车键可取消未输入完成的命令，并退回到 > －提示符。

2.5.2 MongoDB客户端

能运行任意 JavaScript 程序听上去很酷，不过 shell 的真正强大之处在于，它是一个独立的 MongoDB 客户端。启动时，shell 会连到 MongoDB 服务器的 test 数据库，并将数据库连接赋值给全局变量 db。这个变量是通过 shell 访问 MongoDB 的主要入口点。

如果想要查看 db 当前指向哪个数据库，可以使用 db 命令：

```
> db
test
```

为了方便习惯使用 SQL shell 的用户，shell 还包含一些非 JavaScript 语法的扩展。这些扩展并不提供额外的功能，而是一些非常棒的语法糖。例如，最重要的操作之一为选择数据库：